LAMPIRAN

Lampiran 1 Surat Ketersediaan Pembimbing I

SURAT KESEDIAAN MEMBIMBING TA

Yang bertandatangan di bawah ini:

Nama	: Miftakhul Huda, M.Kom
NIDN	: 0620127801
NIPY	: 04.007.033
Jabatan Struktural	: -
Jabatan Fungsional	: Lektor

Dengan ini menyatakan bersedia untuk menjadi pembimbing I pada Tugas Akhir mahasiswa berikut:

Nama	: Iman Sugiro
NIM	: 23041037
Program Studi	: D-3 Teknik Komputer
Judul TA	: PENGEMBANGAN SISTEM PEMANTUAN KEAMANAN GUDANG ELEKTRONIK MENGGUNAKAN LASER DAN ESP32 CAM BERBASIS IOT

Demikian pernyataan ini dibuat agar dapat dilaksanakan sebagaimana mestinya.

Tegal, 29 April 2026

Mengetahui
Ka Prodi DIII Teknik Komputer,

Arif Rakhman, SE, S.Pd, M.Kom.
NIPY. 05.016.291

Dosen Pembimbing I,

Miftakhul Huda, M.Kom
NIPY. 04.007.033

Lampiran 2 Surat Ketersediaan Pembimbing II

SURAT KESEDIAAN MEMBIMBING TA

Yang bertandatangan di bawah ini:

Nama	: Achmad Sutanto, S.Kom, M.Tr.T
NIPY	: 11.012.128
NIDN	: 0618058902
Jabatan Struktural	: Dosen Tetap
Jabatan Fungsional	: Asisten Ahli

Dengan ini menyatakan bersedia untuk menjadi pembimbing II pada Tugas Akhir mahasiswa berikut:

Nama	: Iman Sugiro
NIM	: 23041027
Program Studi	: D-3 Teknik Komputer
Judul TA	: PENGEMBANGAN SISTEM PEMANTUAN KEAMANAN GUDANG ELEKTRONIK MENGGUNAKAN LASER DAN ESP32 CAM BERBASIS IOT

Demikian pernyataan ini dibuat agar dapat dilaksanakan sebagaimana mestinya.

Tegal, 29 April 2026

Mengetahui
Ka Prodi DIII Teknik Komputer,

Arif Rakhman, SE, S.Pd, M.Kom.
NIPY. 05.016.291

Dosen Pembimbing II,

Achmad Sutanto, S.Kom, M.Tr.T
NIPY. 11.012.128

Lampiran 3 Surat Observasi

Universitas
Harkat Negeri

Sekolah Vokasi
Program Studi D-3 Teknik Komputer

Nomor : 566.03/KOM-HN/II/2026
Lampiran : -
Perihal : Permohonan Izin Observasi Tugas Akhir (TA)

Kepada Yth.

Pimpinan Cahaya Elektrik

Di Jl. Teuku Umar No.183,Tegal Selatan,Kota Tegal

Dengan Hormat,

Sehubungan dengan tugas mata kuliah Tugas Akhir (TA) yang akan diselenggarakan di semester V Program Studi Diploma III Teknik Komputer Universitas Harkat Negeri, Maka dengan ini kami mengajukan izin observasi pengambilan data di Cahaya Elektrik yang Bapak / Ibu Pimpin, untuk kepentingan dalam pembuatan produk Tugas Akhir, dengan Mahasiswa sebagai berikut:

No	NIM	NAMA	NO HP
1	23041027	SAEFUL IMRON	085740996048
2	23041037	IMAN SUGIRO	085712776981

Demikian surat permohonan ini kami sampaikan atas izin dan kerjasamanya kami sampaikan terima kasih.

Tegal, 16 Maret 2026
Ka. Prodi DIII Teknik Komputer
Universitas Harkat Negeri

Universitas
Harkat Negeri
DIII Teknik Komputer

Arif Rakhman SE, S.Pd, M.Kom
NIPY. 05.016.291

Program Studi D-3 Teknik Komputer, Sekolah Vokasi
Gedung Utama Ruang B 00 Kampus Mataram - Jalan Mataram No.9 Kota Tegal
WhatsApp: +62 85291833673 | Surel: komputer@harkatnegeri.ac.id

Lampiran 4 Foto Dokumentasi

Lampiran 5 Laporan Buku Bimbingan

Laporan

No	HARI/ TANGGAL	URAIAN	TANDA TANGAN
1	29/4 2026	Bab. I Revisi - Tata tulis - Latar belakang Bab II, III Revisi - Tata tulis	
2	4/5 2026	Acc Bab I, II, III Note: 2 spasi	
3	19/6 2026	Bab IV, V, VI Acc Siap ujian 19/6 2026	

Lampiran 23
Bimbingan Laporan Pembimbing I TA

IK	P2M	PHB	d.5.1.e.1

PEMBIMBING II: *BIMBINGAN LAPORAN TA*

No	HARI/ TANGGAL	URAIAN	TANDA TANGAN
1	26/5 2026	- Memberarkan paragraf - Kata asing - teks Tipe	
2	3/6 2026	- Kata asing - di~~pe~~kalimat kurang lengkap - 1x enter - Diagram blok - Paragraf	
3.	4/6 2026	- Before - after - kata asing -	
4	5/6 2026	- Shif enter - kalimat ada yangg harus perbaiki - Pembenaran Istilah asing	
5.	8/6 2026	- ~~Short~~ flowchart ~~ringgu-nomor-et~~ perbaikan APC acc bab IV-VI perbaiki penulisan, lengkapi tiap bagian Tegal, 13/6 2020 Moh. Sutarno	

Lampiran 6 Source Code

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClient.h>
#include <SoftwareSerial.h>
#include "DHT.h"
#include <WiFiManager.h>

// ================================================
// PIN
// ================================================
#define DHTPIN        D5
#define BUZZER_PIN    D6
#define PIR_PIN       D1
#define LDR_PIN       D2
#define RESET_PIN     D7
#define FLAME_PIN     D0     // Sensor api (flame module) - output
digital
#define DHTTYPE       DHT22

// ================================================
// KONFIGURASI
// ================================================
#define BATAS_SUHU_MAKS       33.8
#define JEDA_BACA_DHT         10000UL   // 10 detik
#define JEDA_KIRIM_NORMAL     300000UL  // 5 menit
#define JEDA_KIRIM_DARURAT    60000UL   // 1 menit saat overheat
#define JEDA_DETEKSI_PIR      8000UL    // cooldown setelah PIR
aktif
#define JEDA_DETEKSI_LASER    8000UL    // cooldown setelah laser
putus
#define JEDA_DETEKSI_API      5000UL    // cooldown setelah api
terdeteksi
#define JEDA_KIRIM_ALARM_API  60000UL   // jeda kirim ulang alarm
api ke server
#define FLAME_ACTIVE_LOW      true      // true jika modul flame
output LOW saat ada api
#define DELAY_BOOTING         15000UL   // tunggu sensor stabil
setelah boot
#define WIFI_RESET_HOLD_MS    5000UL
#define HTTP_TIMEOUT_MS       5000

// Handshake ESP32-CAM
#define CAM_BAUD              9600      // SoftwareSerial di 9600
#define CAM_READY_TIMEOUT     3000UL   // tunggu reply "READY"
dari CAM
#define CAM_PING_INTERVAL     5000UL   // cek status CAM tiap 5
detik

// ================================================
// BUZZER NON-BLOCKING
// ================================================
struct BuzzerPattern {
  int    frekuensi[6];   // durasi ON tiap nada (ms), 0 = selesai
  int    jeda[6];        // jeda antar nada (ms)
```

```
  int    jumlah;
  int    indeks;
  unsigned long waktuMulai;
  bool   sedangBunyi;
};

BuzzerPattern buzzer;

// Daftarkan pola bunyi
void buzzerMulai(int* pola, int* jeda, int jumlah) {
  for (int i = 0; i < jumlah; i++) {
    buzzer.frekuensi[i] = pola[i];
    buzzer.jeda[i]      = jeda[i];
  }
  buzzer.jumlah      = jumlah;
  buzzer.indeks      = 0;
  buzzer.waktuMulai  = millis();
  buzzer.sedangBunyi = true;
  digitalWrite(BUZZER_PIN, HIGH);
}

// Dipanggil tiap loop() — tanpa blocking
void buzzerUpdate() {
  if (!buzzer.sedangBunyi) return;

  unsigned long sekarang = millis();
  int idx = buzzer.indeks;

  if (idx >= buzzer.jumlah) {
    // Selesai
    digitalWrite(BUZZER_PIN, LOW);
    buzzer.sedangBunyi = false;
    return;
  }

  // Fase ON
  if (sekarang - buzzer.waktuMulai < (unsigned
long)buzzer.frekuensi[idx]) return;

  digitalWrite(BUZZER_PIN, LOW);

  if (sekarang - buzzer.waktuMulai < (unsigned
long)(buzzer.frekuensi[idx] + buzzer.jeda[idx])) return;
  buzzer.indeks++;
  buzzer.waktuMulai = millis();

  if (buzzer.indeks < buzzer.jumlah) {
    digitalWrite(BUZZER_PIN, HIGH);
  }
}

// Pola preset
void buzzerBooting()  { int p[]={300};    int
j[]={0};            buzzerMulai(p,j,1); }
void buzzerPIR()      { int p[]={100,100};int
j[]={50,0};         buzzerMulai(p,j,2); }
```

```
void buzzerLaser()    { int p[]={70,70,70};int
j[]={50,50,0};     buzzerMulai(p,j,3); }
void buzzerOverheat() { int p[]={150};    int
j[]={0};            buzzerMulai(p,j,1); }
void buzzerReset()    { int p[]={200,200,200};int
j[]={100,100,0};buzzerMulai(p,j,3); }
void buzzerApi()      { int p[]={60,60,60,60,60,60};int
j[]={40,40,40,40,40,0};buzzerMulai(p,j,6); } // pola cepat &
panjang = darurat kebakaran

// ================================================
// VARIABEL GLOBAL
// ================================================
char serverIpInput[40] = "http://silacahaya.my.id";
bool shouldSaveConfig  = false;

SoftwareSerial espCamSerial(D3, D4);  // RX=D3, TX=D4
DHT dht(DHTPIN, DHTTYPE);

float suhu                    = 0;
float kelembapan              = 0;
float suhuTerakhirKirim       = -999.0;
float kelembapanTerakhirKirim = -999.0;

unsigned long waktuTerakhirDHT          = 0;
unsigned long waktuTerakhirKirimAPI     = 0;
unsigned long waktuTerakhirKirimDarurat = 0;
unsigned long waktuTerakhirPIR          = 0;
unsigned long waktuTerakhirLaser        = 0;
unsigned long waktuTerakhirApi          = 0;
unsigned long waktuTerakhirKirimAlarmApi= 0;
unsigned long waktuTerakhirPingCam      = 0;
unsigned long bootTime                  = 0;

unsigned long resetButtonStart = 0;
bool          resetButtonPressed = false;

// Status koneksi ESP32-CAM
bool camReady = false;

// ================================================
// URL HELPER
// ================================================
String getApiUrl() {
  return String(serverIpInput) + "/data/api_simpan_sensor";
}

// ================================================
// CEK STATUS ESP32-CAM via Serial
// Kirim "PING", tunggu reply "READY"
// ================================================
bool cekStatusCam() {
  while (espCamSerial.available()) espCamSerial.read();

  espCamSerial.println("PING");
```

```
  unsigned long mulai = millis();
  String reply = "";

  while (millis() - mulai < CAM_READY_TIMEOUT) {
    if (espCamSerial.available()) {
      char c = espCamSerial.read();
      reply += c;
      if (reply.indexOf("READY") >= 0) return true;
    }
    yield(); // Cegah watchdog reset
  }

  return false;
}

// ================================================
// KIRIM CAPTURE KE ESP32-CAM (dengan cek status)
// ================================================
void triggerCapture(const char* sumber) {
  if (!camReady) {
    Serial.println("[CAM] Tidak konek, capture dibatalkan (" +
String(sumber) + ")");
    return;
  }

  Serial.println("[CAM] Mengirim CAPTURE dari: " +
String(sumber));
  espCamSerial.println("CAPTURE:" + String(sumber));
}

// ================================================
// KIRIM DATA SENSOR KE API
// ================================================
void kirimDataKeApi(float s, float k, const char* alarm = "") {
  if (WiFi.status() != WL_CONNECTED) {
    Serial.println("[API] WiFi tidak terhubung, skip kirim.");
    return;
  }

  WiFiClient client;
  HTTPClient http;
  http.begin(client, getApiUrl());
  http.setTimeout(HTTP_TIMEOUT_MS);
  http.addHeader("Content-Type", "application/json");

  String json = "{\"suhu\":" + String(s, 1) + ",\"kelembapan\":" +
String(k, 1);
  if (strlen(alarm) > 0) {
    json += ",\"alarm\":\"" + String(alarm) + "\"";
  }
  json += "}";
  int code = http.POST(json);

  if (code > 0) {
    Serial.println("[API] Terkirim: HTTP " + String(code));
    suhuTerakhirKirim       = s;
```

```
    kelembapanTerakhirKirim = k;
  } else {
    Serial.println("[API] Gagal: " + http.errorToString(code));
  }

  http.end();
}

// ================================================
// CEK SENSOR API (FLAME MODULE)
// ================================================
bool cekApiTerdeteksi() {
  int nilai = digitalRead(FLAME_PIN);
  return FLAME_ACTIVE_LOW ? (nilai == LOW) : (nilai == HIGH);
}

// ================================================
// RECONNECT WIFI
// ================================================
void cekRekonekWifi() {
  if (WiFi.status() == WL_CONNECTED) return;

  Serial.println("[WiFi] Putus! Mencoba reconnect...");
  WiFi.reconnect();

  unsigned long mulai = millis();
  while (WiFi.status() != WL_CONNECTED && millis() - mulai <
10000) {
    delay(500);
    Serial.print(".");
    yield();
  }

  if (WiFi.status() == WL_CONNECTED) {
    Serial.println("\n[WiFi] Reconnect berhasil: " +
WiFi.localIP().toString());
  } else {
    Serial.println("\n[WiFi] Reconnect gagal, akan coba di loop
berikutnya.");
  }
}

// ================================================
// WIFI MANAGER CALLBACK
// ================================================
void saveConfigCallback() {
  shouldSaveConfig = true;
}

// ================================================
// SETUP
// ================================================
void setup() {
  Serial.begin(115200);
  espCamSerial.begin(CAM_BAUD);
```

```
  pinMode(BUZZER_PIN, OUTPUT);
  pinMode(RESET_PIN,  INPUT_PULLUP);
  pinMode(PIR_PIN,    INPUT);
  pinMode(LDR_PIN,    INPUT_PULLUP);
  pinMode(FLAME_PIN,  INPUT_PULLUP); // modul flame biasanya open-
collector, aktif LOW saat ada api
  digitalWrite(BUZZER_PIN, LOW);

  dht.begin();

  // --- WiFiManager ---
  WiFiManager wifiManager;
  wifiManager.setSaveParamsCallback(saveConfigCallback);

  WiFiManagerParameter custom_ip("server_ip", "Alamat IP Server",
serverIpInput, 40);
  wifiManager.addParameter(&custom_ip);

  if (!wifiManager.autoConnect("CONFIG_ESP8266", "12345678")) {
    Serial.println("[WiFi] Gagal konek, restart...");
    delay(1000);
    ESP.restart();
  }

  WiFi.setSleep(false);

  if (shouldSaveConfig) {
    strcpy(serverIpInput, custom_ip.getValue());
  }

  Serial.println("[WiFi] Terhubung: " +
WiFi.localIP().toString());
  Serial.println("[INFO] IP Server: " + String(serverIpInput));

  bootTime = millis();

  buzzerBooting();
  Serial.println("[INFO] Menunggu sensor stabil...");
}

// ================================================
// LOOP
// ================================================
void loop() {
  unsigned long sekarang = millis();

  // Update buzzer non-blocking
  buzzerUpdate();

  // ===== TOMBOL RESET WIFI =====
  if (digitalRead(RESET_PIN) == LOW) {
    if (!resetButtonPressed) {
      resetButtonPressed = true;
      resetButtonStart   = millis();
      Serial.println("[RESET] Tombol ditekan, tahan 5 detik...");
    }
```

```
    if (millis() - resetButtonStart >= WIFI_RESET_HOLD_MS) {
      Serial.println("[RESET] Menghapus konfigurasi WiFi...");
      buzzerReset();
      delay(700); // Tunggu buzzer selesai
      WiFiManager wm;
      wm.resetSettings();
      delay(500);
      ESP.restart();
    }
  } else {
    resetButtonPressed = false;
  }

  // ===== CEK KONEKSI WIFI =====
  cekRekonekWifi();

  // ===== PING ESP32-CAM (cek status berkala) =====
  if (sekarang - waktuTerakhirPingCam >= CAM_PING_INTERVAL) {
    waktuTerakhirPingCam = sekarang;
    bool statusBaru = cekStatusCam();

    if (statusBaru != camReady) {
      camReady = statusBaru;
      Serial.println(camReady ? "[CAM] Terhubung dan siap." :
"[CAM] Tidak merespons.");
    }
  }

  // ===== MONITORING SUHU & KELEMBAPAN =====
  if (sekarang - waktuTerakhirDHT >= JEDA_BACA_DHT) {
    waktuTerakhirDHT = sekarang;

    float tempS = dht.readTemperature();
    float tempH = dht.readHumidity();

    if (!isnan(tempS) && !isnan(tempH)) {
      suhu       = tempS;
      kelembapan = tempH;
      Serial.printf("[DHT] Suhu: %.1f C | Kelembapan: %.1f%%\n",
suhu, kelembapan);

      // Kirim jika ada perubahan signifikan ATAU sudah 10 menit
      bool adaPerubahan = (abs(suhu - suhuTerakhirKirim) >= 1.0 ||
                           abs(kelembapan -
kelembapanTerakhirKirim) >= 5.0);
      bool sudahLama    = (sekarang - waktuTerakhirKirimAPI >=
JEDA_KIRIM_NORMAL);

      if (adaPerubahan || sudahLama) {
        kirimDataKeApi(suhu, kelembapan);
        waktuTerakhirKirimAPI = sekarang;
      }

      // Overheat: bunyi + kirim darurat tiap 1 menit (tidak
double kirim)
```

```
      if (suhu > BATAS_SUHU_MAKS) {
        if (!buzzer.sedangBunyi) buzzerOverheat();

        if (sekarang - waktuTerakhirKirimDarurat >=
JEDA_KIRIM_DARURAT) {
          waktuTerakhirKirimDarurat = sekarang;
          // Hanya kirim darurat jika kirim normal belum terjadi
barusan
          if (sekarang - waktuTerakhirKirimAPI >= 2000) {
            Serial.println("[OVERHEAT] Mengirim data darurat...");
            kirimDataKeApi(suhu, kelembapan);
            waktuTerakhirKirimAPI = sekarang;
          }
        }
      }

    } else {
      Serial.println("[DHT] Gagal baca sensor!");
    }
  }

  // ===== SISTEM KEAMANAN (aktif setelah booting stabil) =====
  if (sekarang > bootTime + DELAY_BOOTING) {

    bool pirAktif      = (digitalRead(PIR_PIN) == HIGH); // PIR
active-HIGH
    bool laserTerputus = (digitalRead(LDR_PIN) == HIGH); // LDR:
putus = HIGH

    // PIR terdeteksi
    if (pirAktif && (sekarang - waktuTerakhirPIR >=
JEDA_DETEKSI_PIR)) {
      waktuTerakhirPIR = sekarang;
      Serial.println("[PIR] Gerakan terdeteksi!");
      if (!buzzer.sedangBunyi) buzzerPIR();
      triggerCapture("PIR");
    }

    // Laser terputus
    if (laserTerputus && (sekarang - waktuTerakhirLaser >=
JEDA_DETEKSI_LASER)) {
      waktuTerakhirLaser = sekarang;
      Serial.println("[LASER] Sinar terputus!");
      if (!buzzer.sedangBunyi) buzzerLaser();
      triggerCapture("LASER");
    }

    // Api terdeteksi (flame sensor)
    bool apiTerdeteksi = cekApiTerdeteksi();
    if (apiTerdeteksi && (sekarang - waktuTerakhirApi >=
JEDA_DETEKSI_API)) {
      waktuTerakhirApi = sekarang;
      Serial.println("[API/FLAME] Api terdeteksi!");
      buzzerApi(); // alarm kebakaran selalu prioritas, timpa
bunyi lain
      triggerCapture("FLAME");
```

```
      // Kirim notifikasi darurat ke server (tidak spam, jeda 1
menit)
      if (sekarang - waktuTerakhirKirimAlarmApi >=
JEDA_KIRIM_ALARM_API) {
        waktuTerakhirKirimAlarmApi = sekarang;
        Serial.println("[API/FLAME] Mengirim alarm kebakaran ke
server...");
        kirimDataKeApi(suhu, kelembapan, "kebakaran");
        waktuTerakhirKirimAPI = sekarang;
      }
    }
  }
}
```